Nicholas Chan
nipchan@ucsc.edu
6/06/2021
CSE13s Spring '21

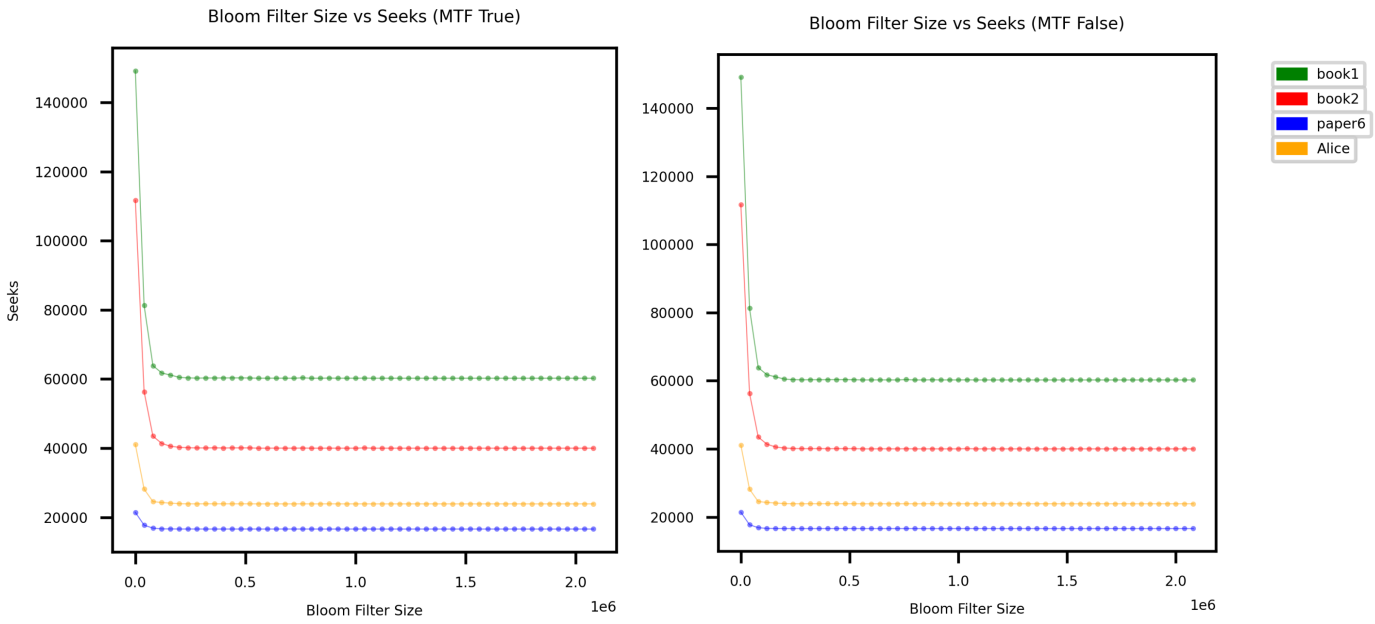
Writeup:
Assignment 7: The Great Firewall of Santa Cruz

For assignment 7, I implemented a firewall for detecting words deemed as "Badspeak" and "Oldspeak" which utilized a bloom filter and hashtable. After completing and testing banhammer.c on text files from the corpora folder in the cse13s-resources repository, I noticed that some relationship existed between the sizes of Bloom filters and hash tables and the number of seeks and average seek lengths. To decipher this relationship, I used a shell script and a modified version of banhammer to print out statistics over many runs to see how altering the sizes of the Bloom filter and hash table affected the number of seeks and average length of seeks performed when running banhammer.

A hypothesis I had going into this experiment was that the number of seeks would increase as the sizes of the Bloom filter or hash table decreased. I also had the hypothesis that average seek lengths would increase as the sizes of the Bloom filter or hash table decreased. I also agreed with the notion that enabling the move to front rule would decrease seeks and average seek lengths as it was a logical assumption considering how linked lists were used in this assignment for handling hash collisions.

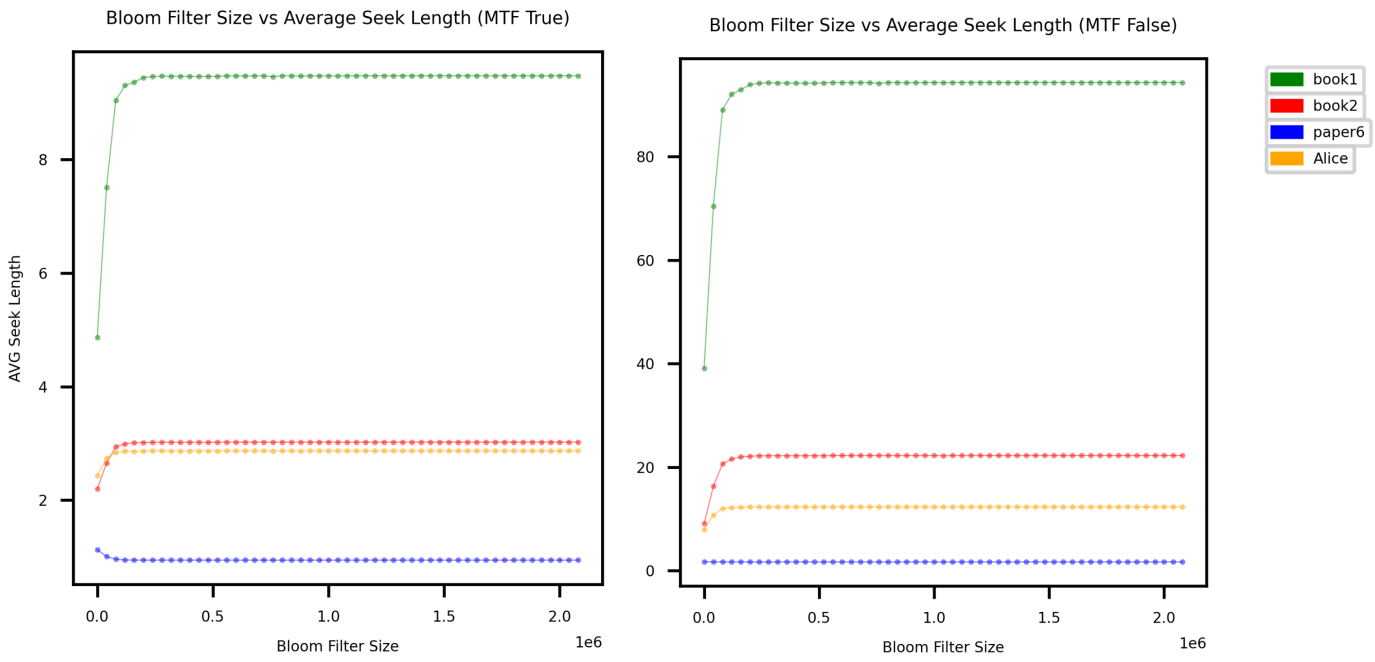
I ran 2 main conditions which each had the condition of enabling or disabling MTF:

- Changing Bloom filter sizes
    - MTF true
    - MTF false
- Changing Hash table sizes
    - MTF true
    - MTF false



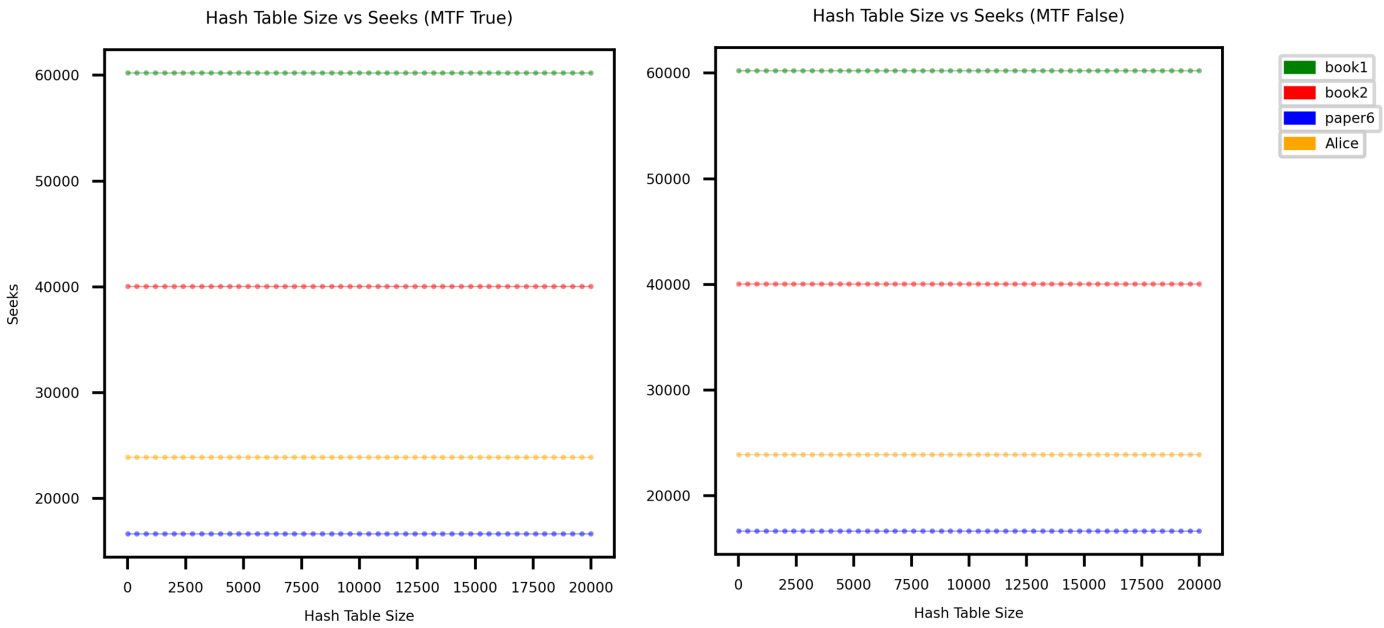
Graphs of Bloom Filter Size vs Seeks. The graph on the left shows the results of enabling the move to front rule while the right shows the results of disabling it

After observing this graph I was pretty surprised to find that the MTF rule didn't have much of an effect when measuring the number of seeks over changing Bloom filter sizes. However the trend that this graph follows seems to agree with my hypothesis that the number of seeks would increase at smaller Bloom Filter sizes. The reasoning for this was that smaller Bloom filters would increase the chances of reporting a false positive, therefore forcing the program to perform a search on the hash table. Something to note is that decreasing Bloom filter size is not the only way to increase the number of seeks in a run. This only makes it so that the possible number of hash keys made in the creation of a Bloom filter is reduced, therefore increasing the likelihood of a false positive result as false positives are defined as a match in the Bloom filter.



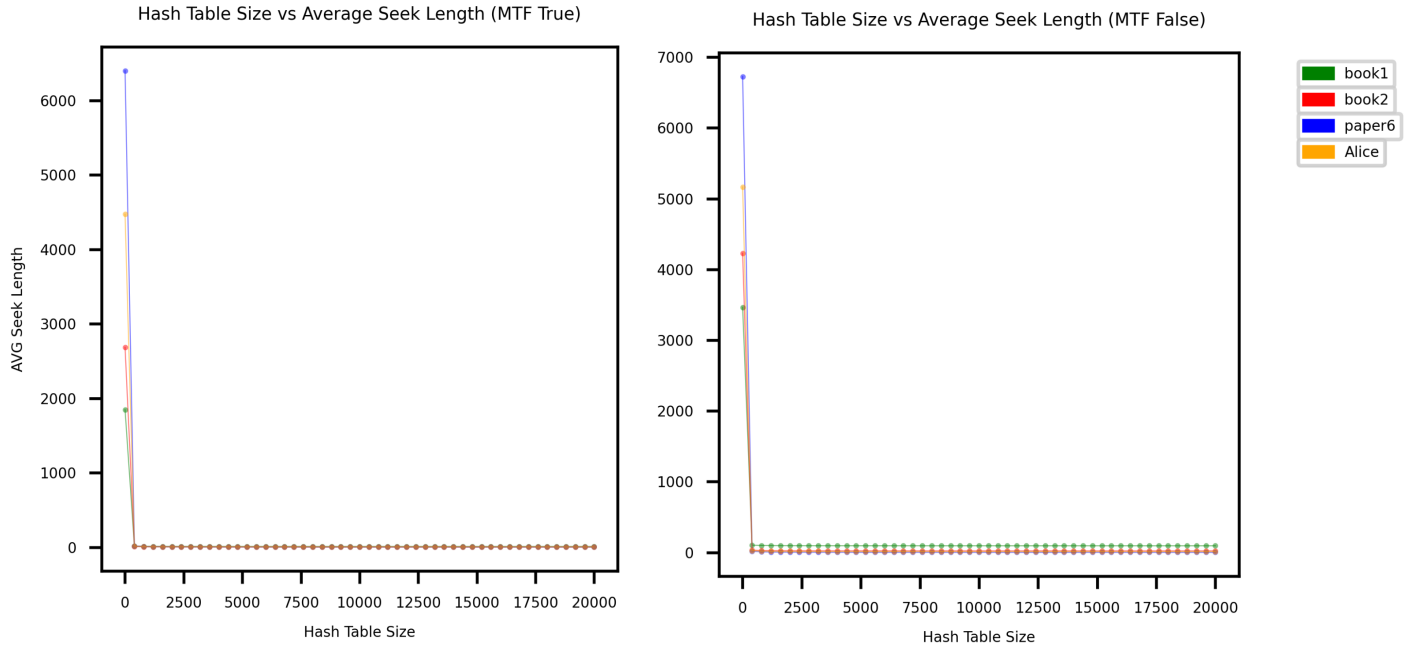
Graphs of Bloom Filter Size vs Average Seek Length. The graph on the left shows the results of enabling the move to front rule while the right shows the results of disabling it

At a first glance of this graph, the effect of enabling MTF can be seen immediately with average seek lengths being a whole order of 10 lower when MTF is enabled than disabled. Coupled with the results from how the total number of seeks changed over Bloom filter size, we can infer that enabling MTF decreases the total amount of links that need to be traversed in a run. The distribution of average seek lengths doesn't seem to differ much between these two conditions, probably because the number of indices available in the hashtable are abundant, providing plenty of room for linked lists and thus reducing the chances of a hash collision in the hash table. Another observation in these graphs is that the smaller the Bloom filter size gets, the lower the average seek length becomes. This is an unusual outcome as paper6 seems to produce the only curve that agrees with my hypothesis that average seek length would increase as bloom filter size decreases. A reasoning for this outcome may be that a lower Bloom filter size increased the number of entries made in the hash table. This increased number of words in the hash table would likely make many short linked list entries, thus skewing our results at lower bloom filter sizes. Average seek lengths seems to normalize as Bloom filter size increases.



Graphs of Hash Table Size vs Seeks. The graph on the left shows the results of enabling the move to front rule while the right shows the results of disabling it

The results from these two graphs illustrating the changes in total number of seeks per run over changing Hash table sizes under either the MTF true or MTF false conditions seem to indicate nothing spectacular about MTF. However these unremarkable results are somewhat consistent with how unremarkable these results were for changing Bloom filter size. There also seems to be no real effect from changing Hash table size.



Graphs of Hash Table Size vs Average Seek Length. The graph on the left shows the results of enabling the move to front rule while the right shows the results of disabling it

There appears to be a slight variation in results when enabling and disabling MTF when observing the change in average seek length over changing hash table size. This variation indicates that enabling MTF can slightly decrease average seek length, at least for small hash table sizes. This result is consistent with the hypothesis I proposed relating to this result, however I did not anticipate such a drop off when approaching larger hash table sizes. This drop off is likely due to a decreased need for traveling linked lists for too long as hash table size increases. As we talked about before, increasing the hash table size increases the possible indices that a hash table can assign indices, thus replacing high linked list lengths with more hash table entries.

After analyzing these results, the functionality of the Bloom filter and hash table became a little more clear to men. Something I may want to consider in the future may be testing text files where the number of repeated words or words that hash to the same position occur often. These conditions seem likely to dramatize the distribution of data considering what I covered in my post assignment analysis. All of my inputs were relatively random and non-repeating. The only difficulty I see with using input with many repetitions may be that too much of my input is coincidentally goodspeak. This could adversely affect my data as I would have practically nothing to graph or work with as they will not even pass my bloom filter. However this condition may also just dramatize the effects of changing Bloom filter size. There are definitely a lot of things to consider when using the Bloom filter and a hash table of linked lists together.